

## 3.3 vs 5 » JeeLabs

[Originalartikel](#)

[Backup](#)

One of the decisions made early on for the JeeNode, was to make it run at 3.3V, instead of the 5V used by the standard Arduino. The main reason for this was the RFM12B wireless module, which can only be used with supply voltages up to 3.8V, according to the specs. Running them at 5V seems to give varying results: I've never damaged one, but there have been reports of such failures. Given that the older RFM12 (no B) worked up to 5V, my hunch is that something in the design was found to give problems at the higher voltage. It's just a guess on my part, though.

So what's the deal with 3.3V vs 5V? Well, the first thing to note, is that the ATmega328 used in a 3.3V JeeNode runs at the same 16 MHz frequency as a 5V Arduino does. This [overclocking](http://en.wikipedia.org/wiki/Over-clocking) is out of spec; but



You're not supposed to do this, but in my experience the good folks at Atmel (the designers and manufacturers of ATmega's and other goodies) have drawn up specifications which are clearly on the conservative side. So much so, that not a single case has been reported where this has caused problems in any of the several thousand JeeNodes produced so far. As I pointed out in a [previous post](https://jeelabs.org/2009/02/27/out-of-spec/index.html), that doesn't necessarily mean everything is 100% perfect over the entire temperature range. But again: no known problems to date. None.

This is good news for low-power uses, BTW. It means you can get the same amount of work done using less power, since power = voltage x current. Even more so because both voltage and current are lower at 3.3V than when running at 5V.

A second reason for running at 3.3V, is that you can use 3 AA batteries instead of 4 (either alkaline or rechargeable). And that you can also power 3.3V circuits with LiPo packs, which have this hugely convenient 3.5..4.2V range.

The third important reason to run JeeNodes at 3.3V, is that more and more neat sensor chips are only available for use in the 2.7 .. 3.6V range or so. By having the entire setup operate at 3.3V, all these sensors can be used without any tedious level converters.

Occasionally I've been bitten by the fact that I used a chip which doesn't work as low as 3.3V, as in the first [RTC Plug](https://jeelabs.org/2009/09/20/clock-plug-v1/index.html) trial. But more often than not, it's simply a matter of looking for alternative chip brands. One recent example was the [555 oscillator](https://jeelabs.org/2010/10/10/generating-ir-pulses/index.html) used on the [Infrared Plug](https://jeelabs.org/ir1.html): the original NE555 needs at least 4.5V, but there's an ICM7555 using CMOS technology which works down to 3V, making it a non-issue.

**Mixing 3.3V and 5V devices**

The trouble with these voltage differences, is not just that the power supply needs to be different. That's the easy bit, since you can always generate 3.3V from a 5V supply with a simple voltage regulator and 2 little capacitors.

The real problem comes from the I/O interface. Placing a 5V signal on a chip running at 3.3V will cause problems, in the worst case even permanently damaging the chip. So each I/O pin connected is also affected by this.

Fortunately, there's often a very simple workaround, using just an extra resistor of 1 k $\Omega$ ; or so in series. To see how this works, here's the way many chips have their input signals hooked up, internally:



There's a pair of diodes inside the chip, for each

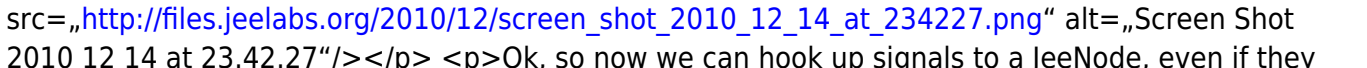
pin (not just the inputs), used for

ESD protection, i.e. to protect the chip against static electricity when you pick it up.

These diodes deflect voltage levels which are above the VCC of the device or below GND level. They do nothing else in normal use, but if you were to place 5V on in a pin of such a device powered by 3.3V, then that would lead to a (potentially large) current through the upper diode.

With electronics (as with humans, btw), it's usually not the voltage itself which causes damage, but the current flow it leads to, and in the case of sensitive electronics components the heat produced from it.

By placing a 1 k $\Omega$  resistor in series, we limit the flow through the diode to under 2 mA, which most devices will handle without any problems:



Ok, so now we can hook up signals to a JeeNode, even if they swing in the 0..5V range. This works best with slow signals, BTW. The extra resistor has a bad effect on rise and fall times of the signal, so don't expect this to work with signals which are in the 1 MHz range or higher. Then again, it's unlikely you'll need to tie such fast signals directly to an ATmega anyway.

How about the other direction?

What if you have a chip running at 5V which needs to receive signals from a chip running at 3.3V, i.e. signals going in the other direction?

Well, it turns out that this may or may not work by simply tying the two lines together. The 3.3V output signal will definitely not damage a chip running at 5V. The worst that can happen, is that the 5V side doesn't consider the signal valid.

We need to look into logic levels to figure this one out, as specified in the datasheet of the chip. The easy part is logic 0, i.e. a low level. Most chips consider anything between 0 and 0.8V a logic low. There will hardly ever be an issue when tying a 3.3V chip to a 5V chip.

The tricky part is logic 1, i.e. a signal which is intended to represent a high level. Now it all depends on what the 3.3V chip sends out, and what the 5V chip requires.

Most CMOS chips, including the ATmega, send out nearly the full power line voltage to represent a logic 1 (when the load current is low), so you can expect output signals to be just about 3.3V on a JeeNode.

On the input side, there are two common cases. Some chips consider everything above 1.6V or so to be a logic one. These chips will be perfectly happy with the JeeNode signal.

The only case when things may or may not work reliably, is with chips which specify the minimum logic 1 voltage to be  $0.7 \times VCC$  or something like that. On a 5V chip, that translates to a minimum value of 3.5V.

Note that datasheets usually contain conservative specs, meant to indicate limit values under all temperatures, load conditions, supply voltages, etc.

In practice, I find that even with  $0.7 \times VCC$ , I can usually drive a 5V chip just fine from a JeeNode. The only exception being higher power chips, such as stepper motor drivers and such, which operate mostly at much higher voltage levels anyway. For these, you may have to use special level translator chips, or perhaps something like the I2C-based

Output Plug, which can be powered with voltages up to 50V or so.

This post only addresses digital I/O signals. With analog I/O, i.e. varying voltage levels, you will need to carefully review what voltage ranges are generated and expected, and perhaps insert either a voltage divider or an op-amp to amplify voltages. That's a bit more involved.

But all in all, living mostly in a 3.3V world is often more flexible than living mostly in a 5V world, nowadays.

Which is the fourth reason why I decided to run JeeNodes at 3.3V, BTW.

From:

<https://schnipsl.qgelm.de/> - Qgelm

Permanent link:

<https://schnipsl.qgelm.de/doku.php?id=wallabag:3.3-vs-5--jeelabs>

Last update: **2021/12/06 15:24**

